UPES

UNIVERSITY OF TOMORROW

SYSTEM PROVISIONING AND CONFIGURATION MANAGEMENT

LAB FILE

NAME: SMRITI RAI

SAP ID: 500096396

BATCH: B3

SUBMITTED TO: Dr. Hitesh Kumar Sharma

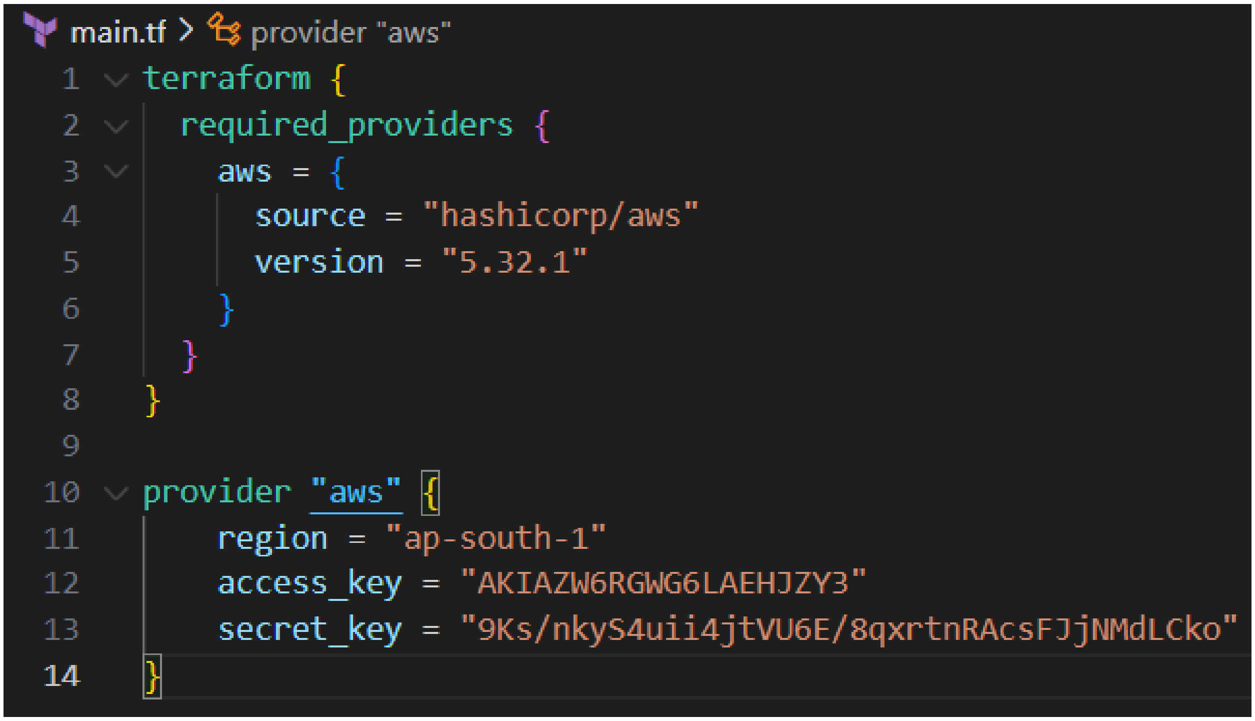
SEMESTER: VI

ENROLLMENT NO.: R2142211212

EXPERIMENT 9:

Creating Multiple EC2 instances with for_each in Terraform

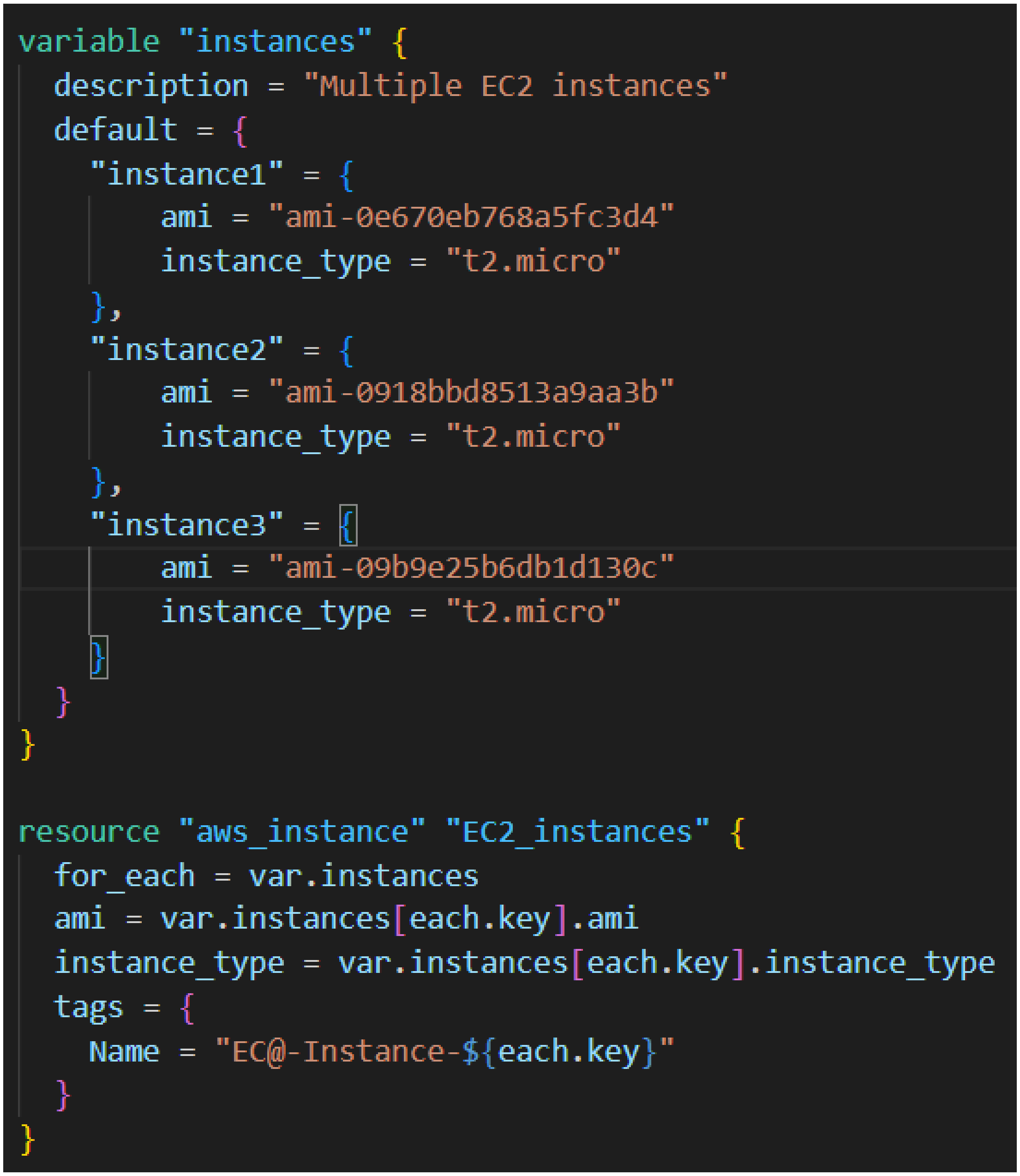
1. Create a filed named main.tf.



main.tf > provider "aws"

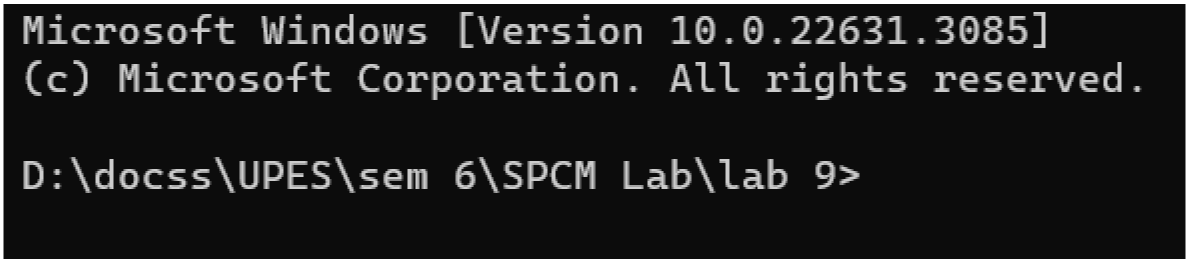
```
terraform {
  required_providers {
    aws = {
      source = "hashicorp/aws"
      version = "5.32.1"
    }
  }
}

provider "aws" {
    region = "ap-south-1"
    access_key = "AKIAZW6RGWG6LAEHJZY3"
    secret_key = "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"
}
```

2. Use for_each to configure 3 EC2 instances.



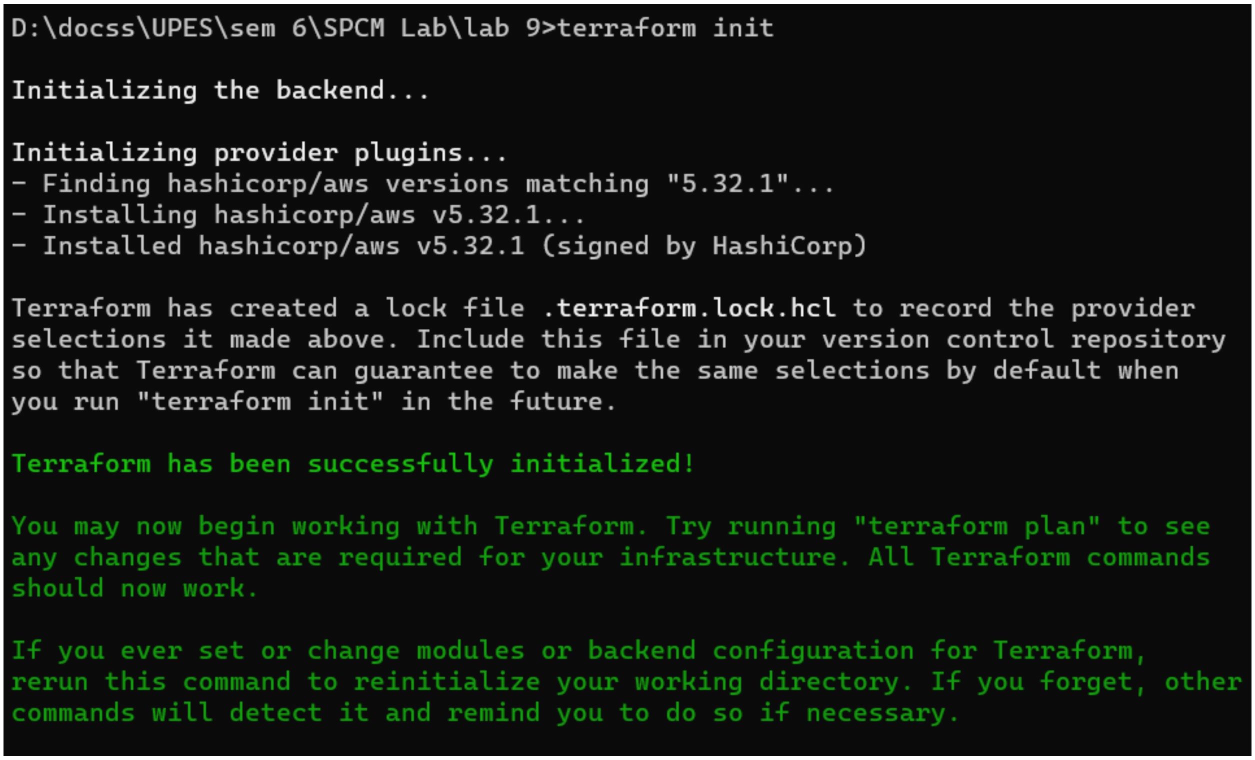
```
variable "instances" {
  description = "Multiple EC2 instances"
  default = {
    "instance1" = {
        ami = "ami-0e670eb768a5fc3d4"
        instance_type = "t2.micro"
    },
    "instance2" = {
        ami = "ami-0918bbd8513a9aa3b"
        instance_type = "t2.micro"
    },
    "instance3" = {
        ami = "ami-09b9e25b6db1d130c"
        instance_type = "t2.micro"
    }
  }
}

resource "aws_instance" "EC2_instances" {
  for_each = var.instances
  ami = var.instances[each.key].ami
  instance_type = var.instances[each.key].instance_type
  tags = {
    Name = "EC@-Instance-${each.key}"
  }
}
```

3. Go to command prompt and navigate to your directory



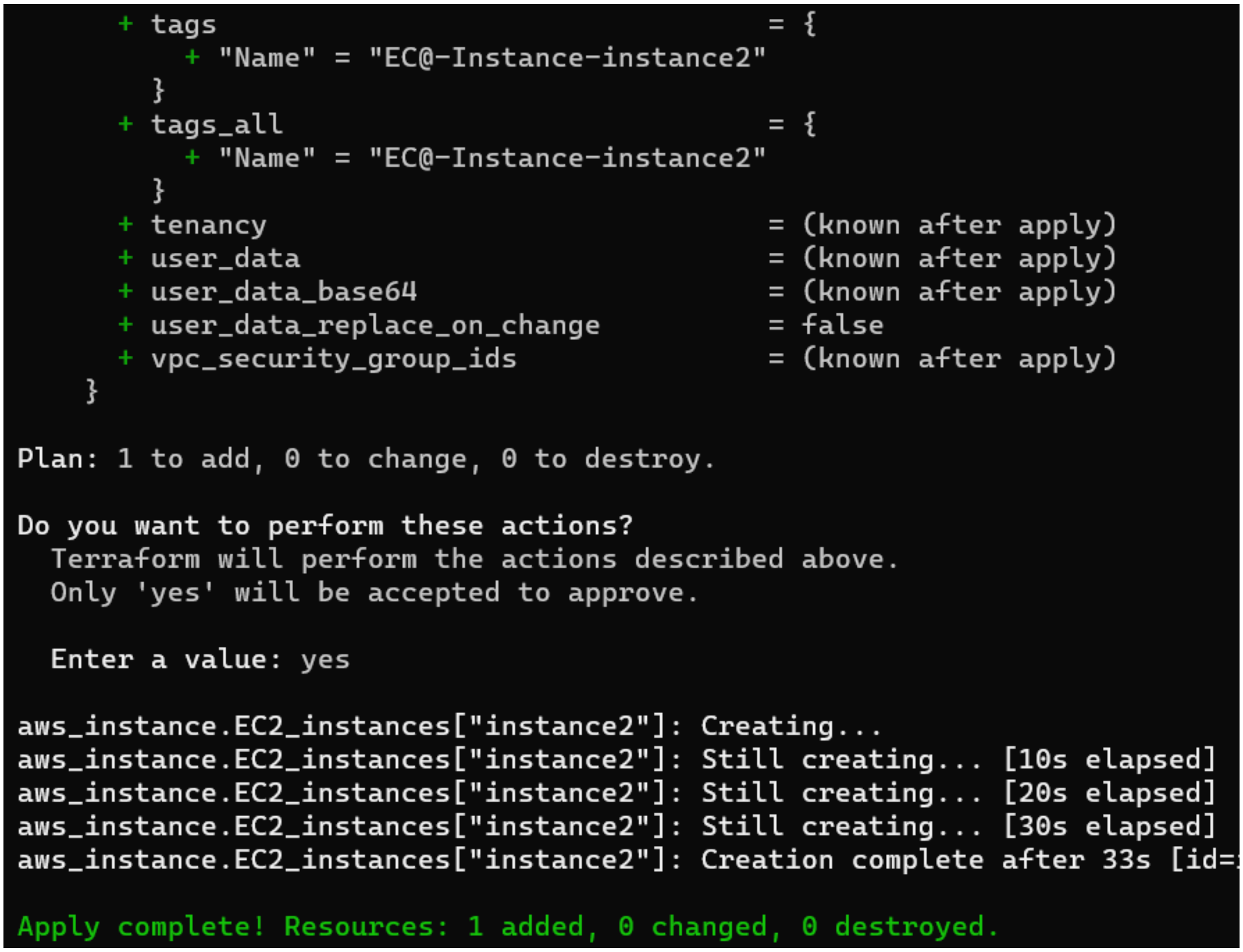
```
Microsoft Windows [Version 10.0.22631.3085]
(c) Microsoft Corporation. All rights reserved.

D:\docss\UPES\sem 6\SPCM Lab\lab 9>
```

4. Initialize the repository using terraform init



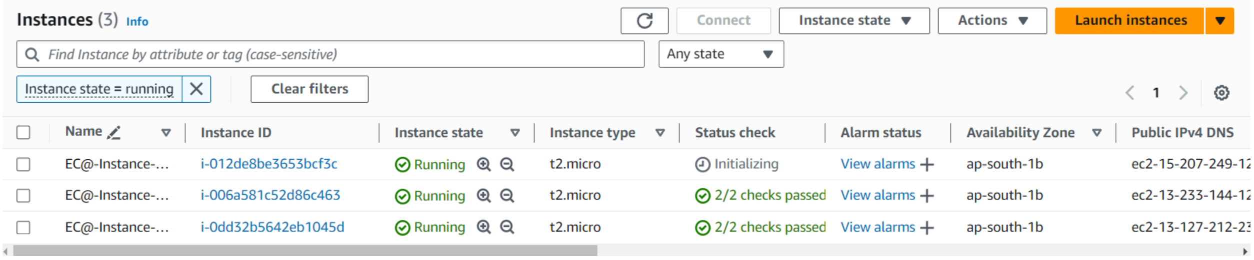
```
D:\docss\UPES\sem 6\SPCM Lab\lab 9>terraform init

Initializing the backend...

Initializing provider plugins...
- Finding hashicorp/aws versions matching "5.32.1"...
- Installing hashicorp/aws v5.32.1...
- Installed hashicorp/aws v5.32.1 (signed by HashiCorp)

Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
```

5. Apply the changes using terraform apply



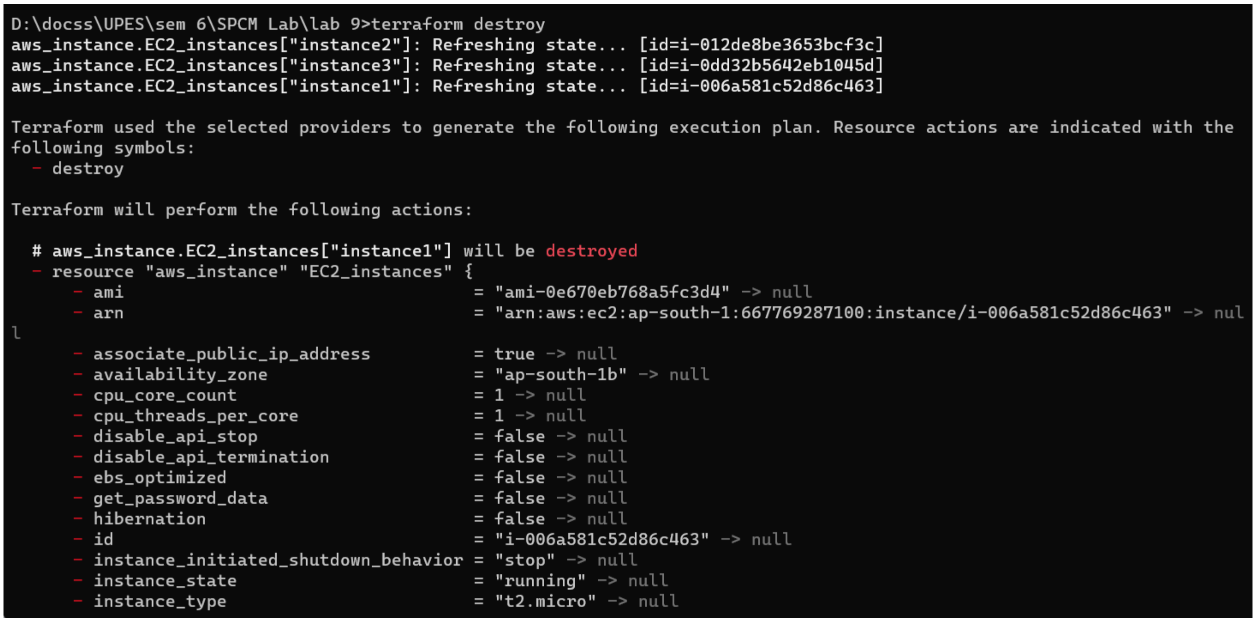
```
      + tags                                 = {
          + "Name" = "EC@-Instance-instance2"
        }
      + tags_all                             = {
          + "Name" = "EC@-Instance-instance2"
        }
      + tenancy                              = (known after apply)
      + user_data                            = (known after apply)
      + user_data_base64                     = (known after apply)
      + user_data_replace_on_change          = false
      + vpc_security_group_ids               = (known after apply)
    }

Plan: 1 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes

aws_instance.EC2_instances["instance2"]: Creating...
aws_instance.EC2_instances["instance2"]: Still creating... [10s elapsed]
aws_instance.EC2_instances["instance2"]: Still creating... [20s elapsed]
aws_instance.EC2_instances["instance2"]: Still creating... [30s elapsed]
aws_instance.EC2_instances["instance2"]: Creation complete after 33s [id=

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
```

6. Go to AWS console to check the creation of your resources.

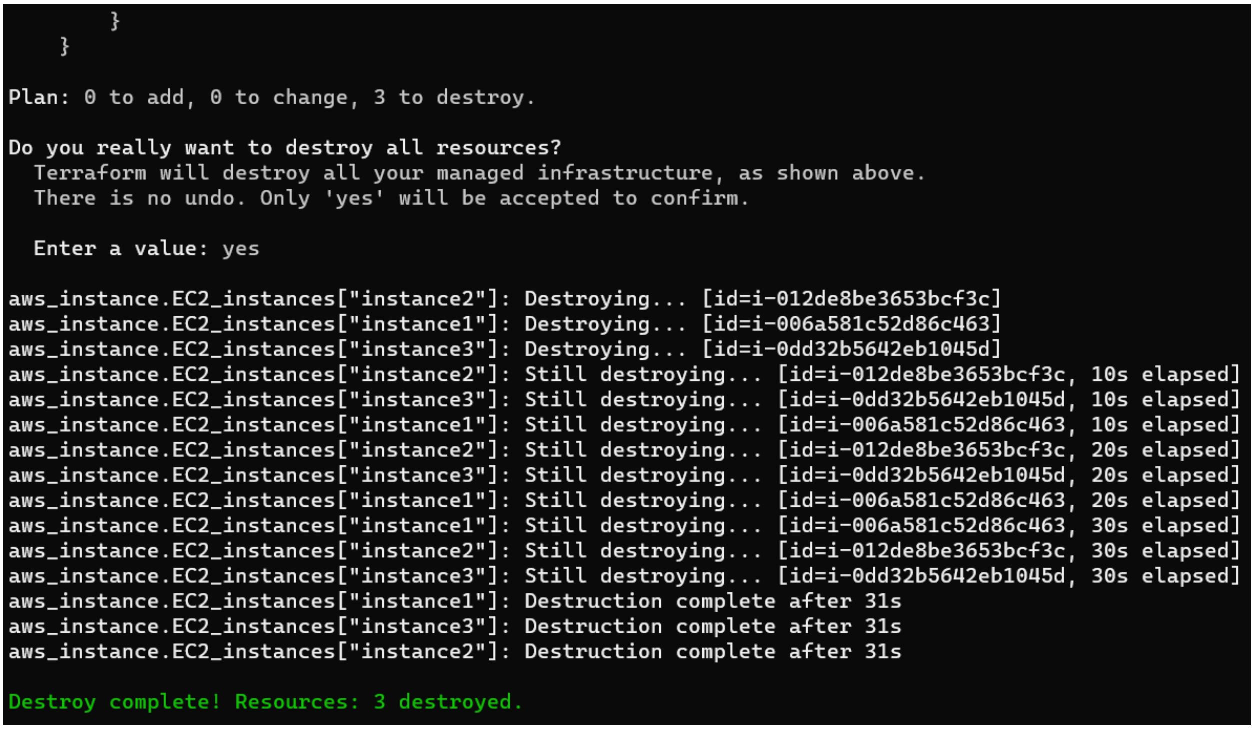


□	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
□	EC@-Instance-…	i-012de8be3653bcf3c	Running	t2.micro	Initializing	View alarms +	ap-south-1b	ec2-15-207-249-12
□	EC@-Instance-…	i-006a581c52d86c463	Running	t2.micro	2/2 checks passed	View alarms +	ap-south-1b	ec2-13-233-144-12
□	EC@-Instance-…	i-0dd32b5642eb1045d	Running	t2.micro	2/2 checks passed	View alarms +	ap-south-1b	ec2-13-127-212-23

7. Clean up your resources using terraform destroy.



```
D:\docss\UPES\sem 6\SPCM Lab\lab 9>terraform destroy
aws_instance.EC2_instances["instance2"]: Refreshing state... [id=i-012de8be3653bcf3c]
aws_instance.EC2_instances["instance3"]: Refreshing state... [id=i-0dd32b5642eb1045d]
aws_instance.EC2_instances["instance1"]: Refreshing state... [id=i-006a581c52d86c463]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the
following symbols:
  - destroy

Terraform will perform the following actions:

  # aws_instance.EC2_instances["instance1"] will be destroyed
  - resource "aws_instance" "EC2_instances" {
      - ami                                  = "ami-0e670eb768a5fc3d4" -> null
      - arn                                  = "arn:aws:ec2:ap-south-1:667769287100:instance/i-006a581c52d86c463" -> nul
l
      - associate_public_ip_address          = true -> null
      - availability_zone                    = "ap-south-1b" -> null
      - cpu_core_count                       = 1 -> null
      - cpu_threads_per_core                 = 1 -> null
      - disable_api_stop                     = false -> null
      - disable_api_termination              = false -> null
      - ebs_optimized                        = false -> null
      - get_password_data                    = false -> null
      - hibernation                          = false -> null
      - id                                   = "i-006a581c52d86c463" -> null
      - instance_initiated_shutdown_behavior = "stop" -> null
      - instance_state                       = "running" -> null
      - instance_type                        = "t2.micro" -> null
```



```
        }
    }

Plan: 0 to add, 0 to change, 3 to destroy.

Do you really want to destroy all resources?
  Terraform will destroy all your managed infrastructure, as shown above.
  There is no undo. Only 'yes' will be accepted to confirm.

  Enter a value: yes

aws_instance.EC2_instances["instance2"]: Destroying... [id=i-012de8be3653bcf3c]
aws_instance.EC2_instances["instance1"]: Destroying... [id=i-006a581c52d86c463]
aws_instance.EC2_instances["instance3"]: Destroying... [id=i-0dd32b5642eb1045d]
aws_instance.EC2_instances["instance2"]: Still destroying... [id=i-012de8be3653bcf3c, 10s elapsed]
aws_instance.EC2_instances["instance3"]: Still destroying... [id=i-0dd32b5642eb1045d, 10s elapsed]
aws_instance.EC2_instances["instance1"]: Still destroying... [id=i-006a581c52d86c463, 10s elapsed]
aws_instance.EC2_instances["instance2"]: Still destroying... [id=i-012de8be3653bcf3c, 20s elapsed]
aws_instance.EC2_instances["instance3"]: Still destroying... [id=i-0dd32b5642eb1045d, 20s elapsed]
aws_instance.EC2_instances["instance1"]: Still destroying... [id=i-006a581c52d86c463, 20s elapsed]
aws_instance.EC2_instances["instance1"]: Still destroying... [id=i-006a581c52d86c463, 30s elapsed]
aws_instance.EC2_instances["instance2"]: Still destroying... [id=i-012de8be3653bcf3c, 30s elapsed]
aws_instance.EC2_instances["instance3"]: Still destroying... [id=i-0dd32b5642eb1045d, 30s elapsed]
aws_instance.EC2_instances["instance1"]: Destruction complete after 31s
aws_instance.EC2_instances["instance3"]: Destruction complete after 31s
aws_instance.EC2_instances["instance2"]: Destruction complete after 31s

Destroy complete! Resources: 3 destroyed.
```